Date
27/01/21

Neeraj Appai

A-J

Practical - 05

Aim - Write a program to implement univariate linear Regression Import dataset of your choice

Description - Univariate linear regression focuses on determining relationship between on independent variable and one dependent variable regression comes handy mainly in situation where the relation between two features is not obvious in naked eye.

Algorithm

1) Start
2) import modules (numpy, csv, matplotlib.pyplot)
3) insert function read_data
4) open file to with csv.reader to read
5) add class linear Regression
6) insert functions __init__, initialize_theta, add_one, cost_function, fit, gradient_descent, predict and compare
7) add plt.plot to display the plot
8) add 'main' function to read csv. file
9) Direct it to linear Regression function
10) Stop

p5_ai (1).py - E:\fffiiles\college pracs and projects\AI\p5_ai (1).py (3.8.3)

File Edit Format Run Options Window Help

```
import numpy as np
import csv
import matplotlib.pyplot as plt
print("Neeraj Appari")
def read_data(filename):
    x, y = list(), list()

    with open(filename, 'r') as csv_file:
        csv_reader = csv.reader(csv_file)

        for row in csv_reader:
            x.append(float(row[0]))
            y.append(float(row[1]))

    x, y = np.array(x), np.array(y)
    print(x)
    print(y)

    return x, y
class LinearRegression:
    def __init__(self, x, y):
        self.x = self.add_ones(x)
        self.y = y
        self.theta = self.initialize_theta()
        self.m = len(y)
    def initialize_theta(self):
        return np.zeros(2)
    def add_ones(self, x):
        return np.array([(1, ele) for ele in x])

    def cost_function(self):
        J = np.sum(np.power((np.dot(self.x, self.t
```

Python 3.8.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.19
25 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more informa
tion.
>>>
======== RESTART: E:\fffiiles\college pracs and projects\AI\p5_ai (
1).py =======
Neeraj Appari
[ 6.1101  5.5277  8.5186  7.0032  5.8598  8.3829  7.4764  8.5781  6
.4862
  5.0546  5.7107 14.164   5.734   8.4084  5.6407  5.3794  6.3654  5
.1301
  6.4296  7.0708  6.1891 20.27    5.4901  6.3261  5.5649 18.945  12
.828
 10.957  13.176  22.203   5.2524  6.5894  9.2482  5.8918  8.2111  7
.9334
  8.0959  5.6063 12.836   6.3534  5.4069  6.8825 11.708   5.7737  7
.8247
  7.0931  5.0702  5.8014 11.7     5.5416  7.5402  5.3077  7.4239  7
.6031
  6.3328  6.3589  6.2742  5.6397  9.3102  9.4536  8.8254  5.1793 21
.279
 14.908  18.959   7.2182  8.2951 10.236   5.4994 20.341  10.136   7
.3345
  6.0062  7.2259  5.0269  6.5479  7.5386  5.0365 10.274   5.1077  5
.7292
  5.1884  6.3557  9.7687  6.5159  8.5172  9.1802  6.002   5.5204  5
.0594
  5.7077  7.6366  5.8707  5.3054  8.2934 13.394   5.4369]
[17.592    9.1302  13.662   11.854    6.8233  11.886    4.3483  12.
  6.5987   3.8166   3.2522  15.505    3.1551   7.2258   0.71618  3.
5129
```

27 August 2021
Friday

Type here to search

ENG 19:57 27-08-2021

p5_ai (1).py - E:\fffiiles\college pracs and projects\AI\p5_ai (1).py (3.8.3)

File Edit Format Run Options Window Help

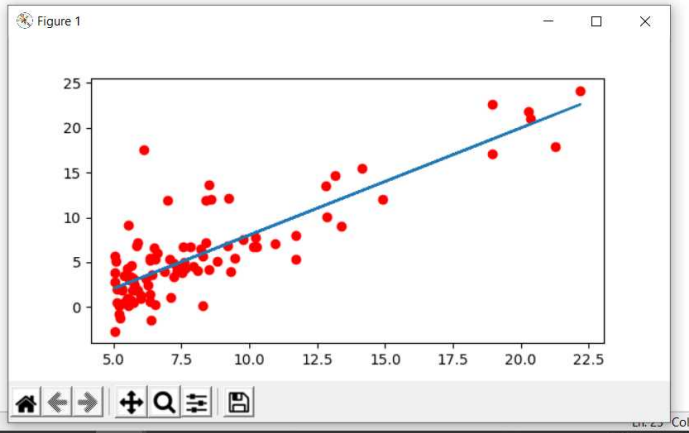
```
        return np.array([(1, ele) for ele in x])

    def cost_function(self):
        J = np.sum(np.power((np.dot(self.x, self.theta) - self.y), 2)) / (2 * self.m)
        return J

    def fit(self, alpha, num_iters):
        self.alpha = alpha
        self.num_iters = num_iters
        self.gradient_descent()
    def gradient_descent(self):
        self.J_history = list()
        for i in range(self.num_iters):
            self.theta = self.theta - (self.alpha / self.m * np.dot((np.dot(self.x, self.theta) - self.y), self.x))
            J = self.cost_function()
            if (i % 100 == 0):
                self.J_history.append(J)

    def predict(self, x):
        x = self.add_ones(x)
        return (np.dot(x, self.theta))

    def compare(self):
        plt.plot(self.x[:, 1], self.y, 'ro')
        plt.plot(self.x[:, 1], np.dot(self.x, self.theta))
        plt.show()
if __name__ == "__main__":
    x, y = read_data('data.csv')
    lr = LinearRegression(x, y)
    lr.fit(alpha= 0.01, num_iters= 15000)
    lr.compare()
```



Ln: 23 Col: 18

Type here to search

ENG 19:58 27-08-2021